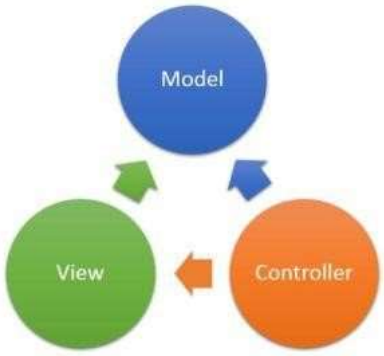
PRIMENA *ENTITY DATA MODEL-A* U RAZVOJU INTERNET ORIJENTISANIH APLIKACIJA PRIMENOM *ASP.NET MVC* ŠABLONA

Osnove ASP.NET MVC

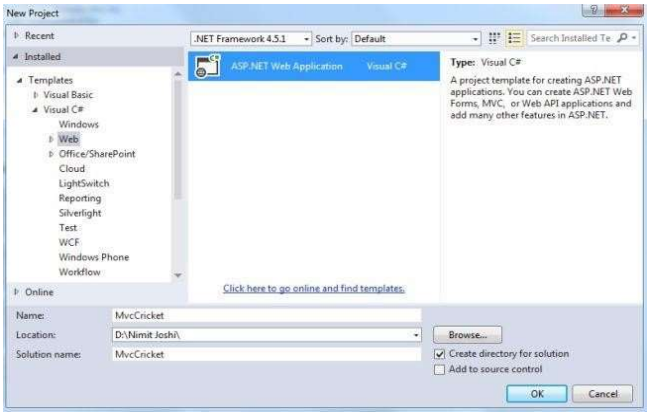
Od verzije Visual Studio 2013 dodate su brojne nove karakteristike između ostalih i MVC. Arhitektura za razvoj aplikacija.



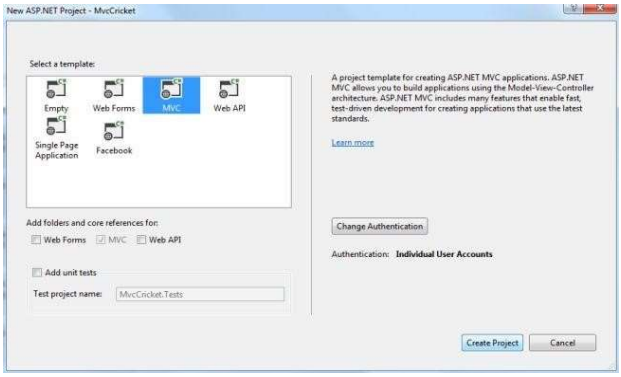
ASP.NET MVC – izrada dinamičkih sajtova vođenih podacima. Uvodi nova svojstva poput SPA – (eng. single page applications), optimizacije za mobilne uređaje i adaptivno renderovanje.

Kreiranje aplikacije i dodavanje kontrolera

Kreirajte aplikaciju

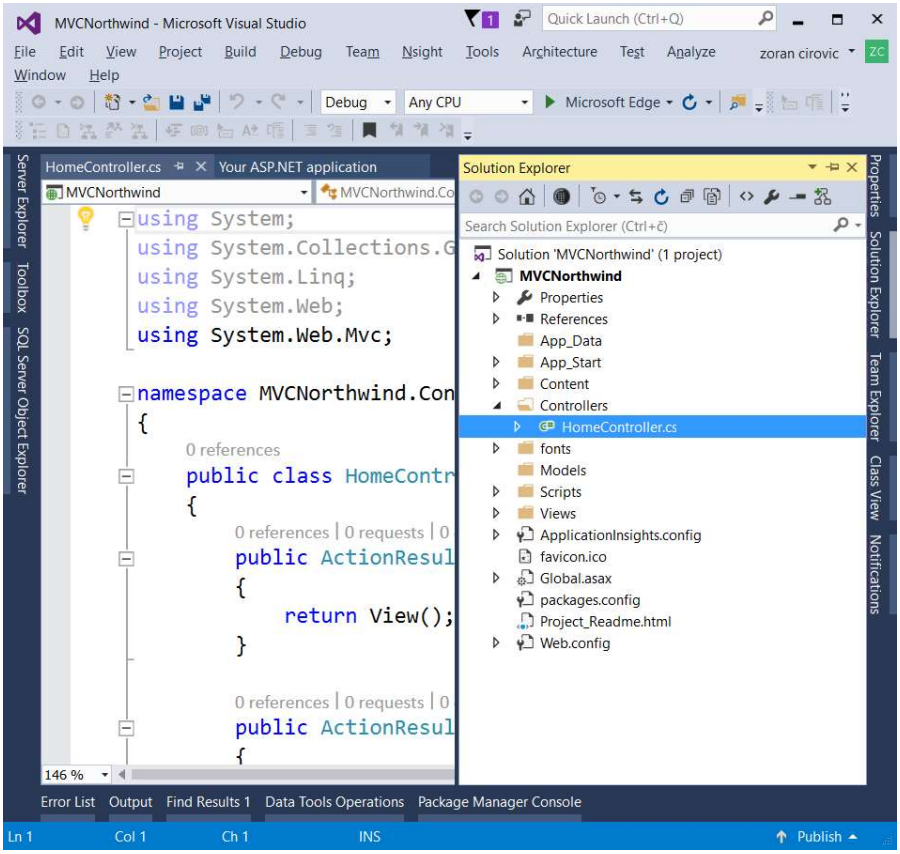


Korak 1: MVC šablon



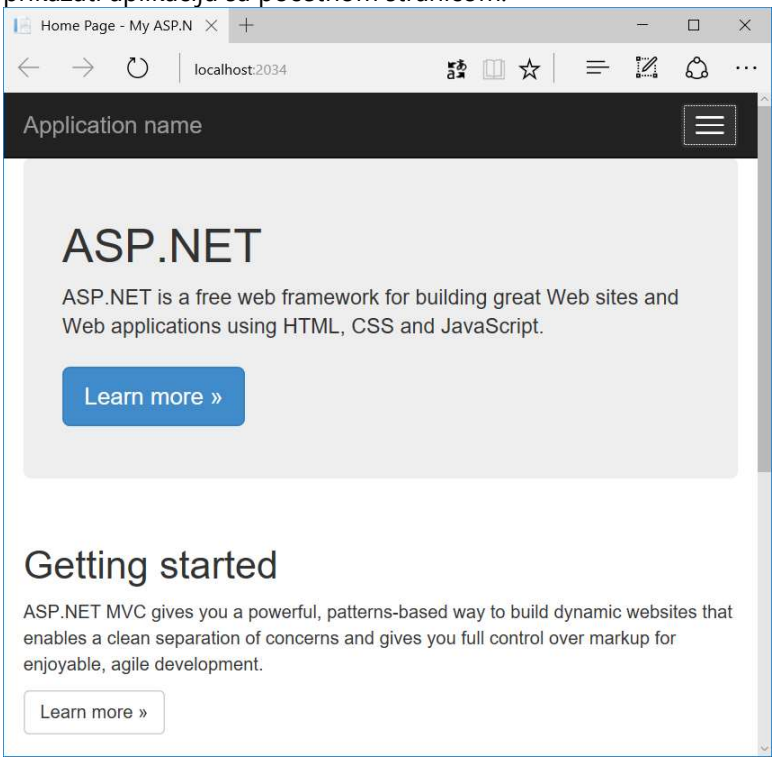
Korak 2: Klik na Create Project.

Pošto izvedete klik na Create Project, Visual Studio automatski kreira MVC web aplikaciju u kojoj možete videti foldere kao što su: **Models**, **Controllers** i **Views**. Postoje dva fajla/foldera Account i Home u folderu Controller odnosno View. Kao što vidite na narednoj slici *HomeController.cs* fajl ima odgovarajući pogled u folderu Views.



Debug Application

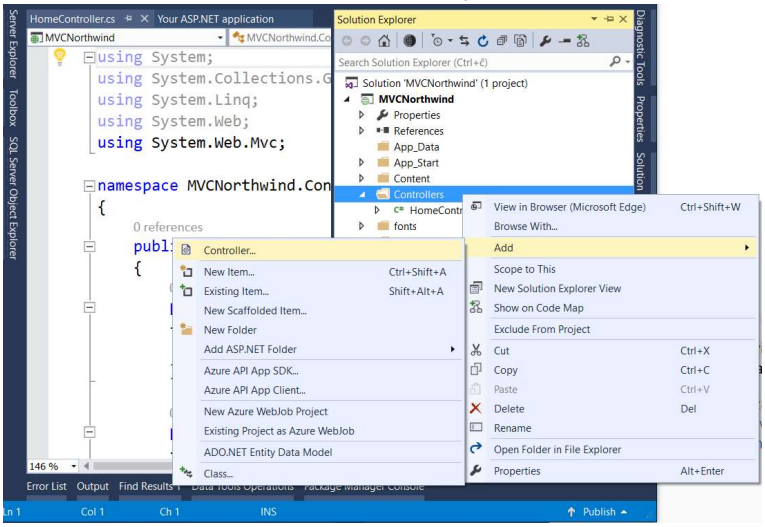
Pritiskom na F5 pokrećete izvršavanje sa debagovanjem. Visual Studio omogućava debagovanje pomoću hostovaja urađenog pomoću IIS Express web servera. Neki od podešenih web čitača će se otvoriti i prikazati aplikaciju sa početnom stranicom.



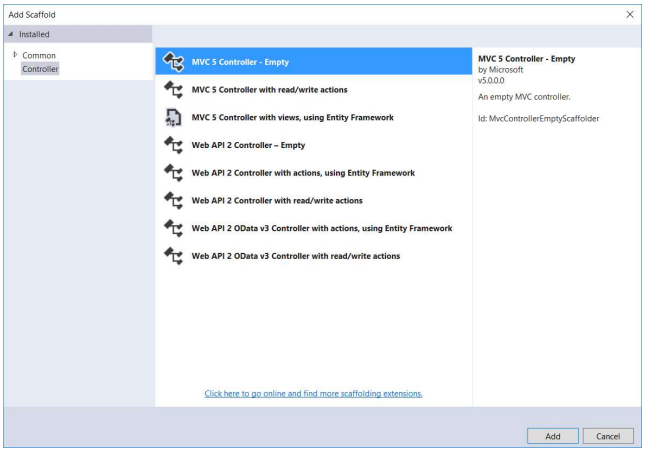
U adres-baru web čitača nalazi se URL i broj porta aplikacije koja se hostuje. *Localhost* zači da se pokrenuli aplikaciju na lokalnom računaru a broj porta je generisan od strane Visual Studio i koristi se pri pokretanju debagovanja.

Rad sa kontrolerom

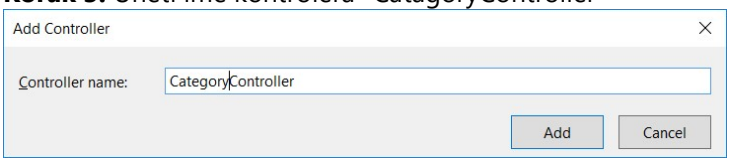
Mada je nekoliko kontrolera već automatski krerirano, mi kreiramo novi:
Korak 1: Desni Klik na folder Controller, a zatim Add->Controller.



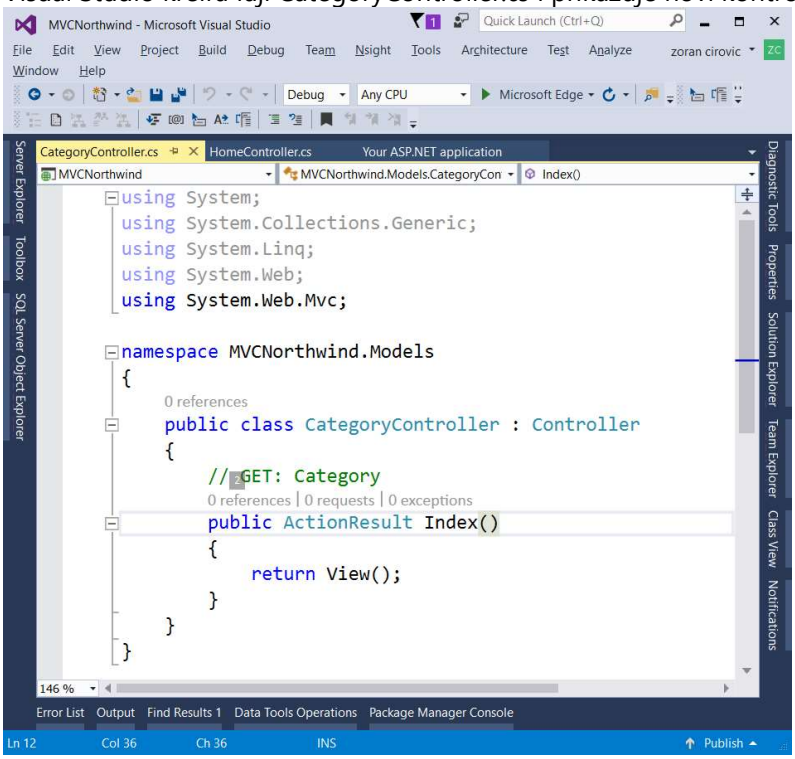
Korak2: Klik na Add Scaffold



Korak 3: Uneti ime kontrolera "CatagoryController"



Visual Studio kreira fajl CategoryController.*cs* i prikazuje novi kontroler.



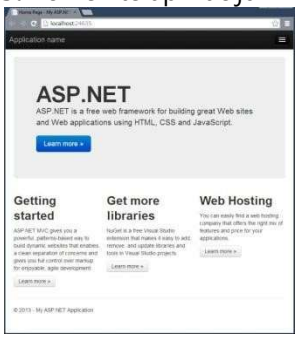
Korak 4: Promeiti kod:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
```

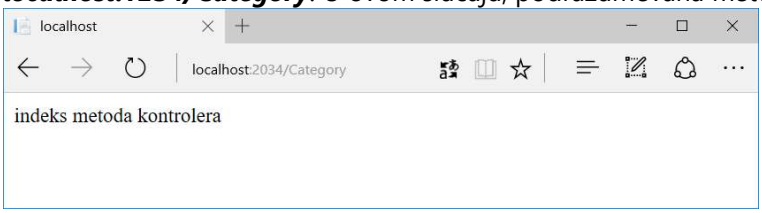
```
namespace MVCNorthwind.Models
{
    public class CategoryController : Controller
    {
        // GET: Category
        public string Index()
        {
            return "indeks metoda kontrolera";
        }

        public string Welcome()
        {
            return "Welcome metoda kontrolera";
        }

    }
}
```

Obratite pažnju na povratnu vrednost metoda.

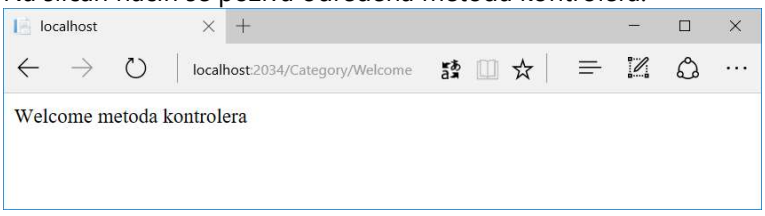
Korak 5: Pokrenite aplikaciju



Podrazumevano Visual Studio otvara HomeController. Moguće je adresirati kontroler na sledeći način: ***localhost:1234/Category***. U ovom slučaju, podrazumevana metoda je Index.



Na sličan način se poziva određena metoda kontrolera:



Dodavanje parametara u akcijama kontrolera

Dodajmo i metodu Browse tako da vrati upitni string na osnovu URL. Na primer da dodaćemo parametar "name" koji će nam obezbediti upit po osnovu vrednosti ovog parametra. Šta treba da uradimo?

```
// GET: /Category/Browse?name=Produce
public string Browse(string name)
```

```
{
    string message =
        HttpUtility.HtmlEncode("Northwind.Category, CategoryName = " + name);
    return message;
}
```

Korisnički unos i primena HTML ENCODING tehnike

U prethodnom primeru koristili smo pomoćnu metodu HttpUtility.HtmlEncode da bi pročistili korisnički ulaz. Na ovaj način štitimo se od JavaScript odnosno od opasne ugradnje HTML koda na našem pogledu. Zamislite sledeći unos:

```
/Category/Browse?Name=<script>window.location='http://hacker.example.com'</script>
```

Šta je rezultat ovog zahteva?

Adresiranje kontrolera

MVC poziva klase kontroler i akcije koje mu pripadaju u zavisnosti od dolazećeg URL. Podrazumevana logika URL rutiranje koristi sledeći format:

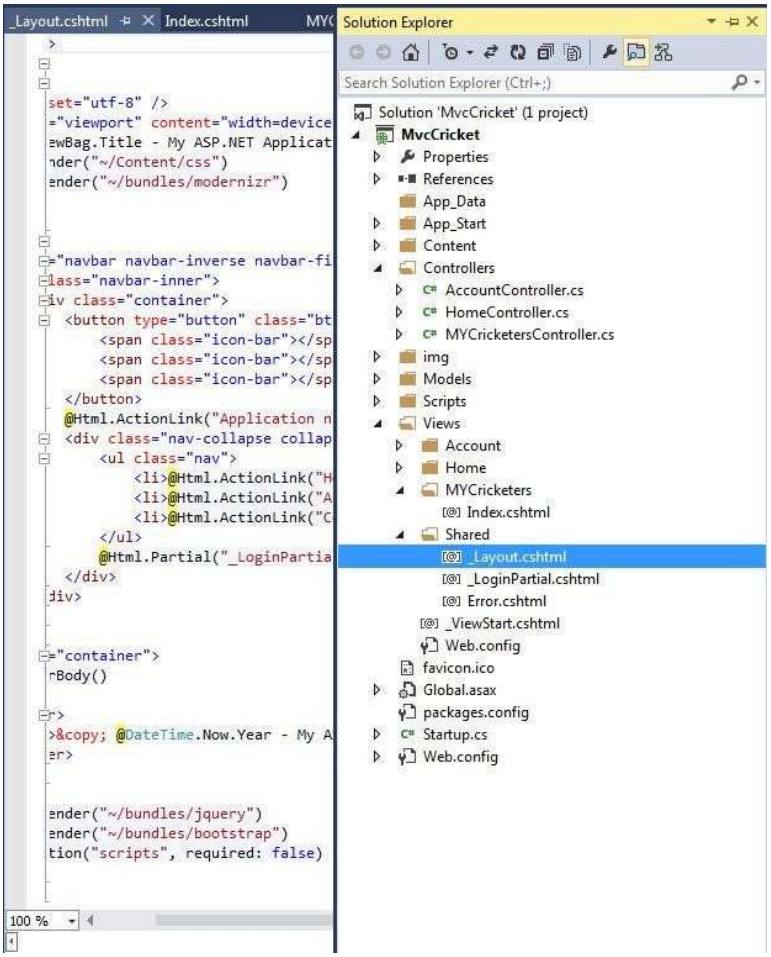
/[Controller]/[ActionName]/[Parameters]

Podešaavanje formata rutiranja se vrši u **RouteConfig.cs** foldera App_Start.

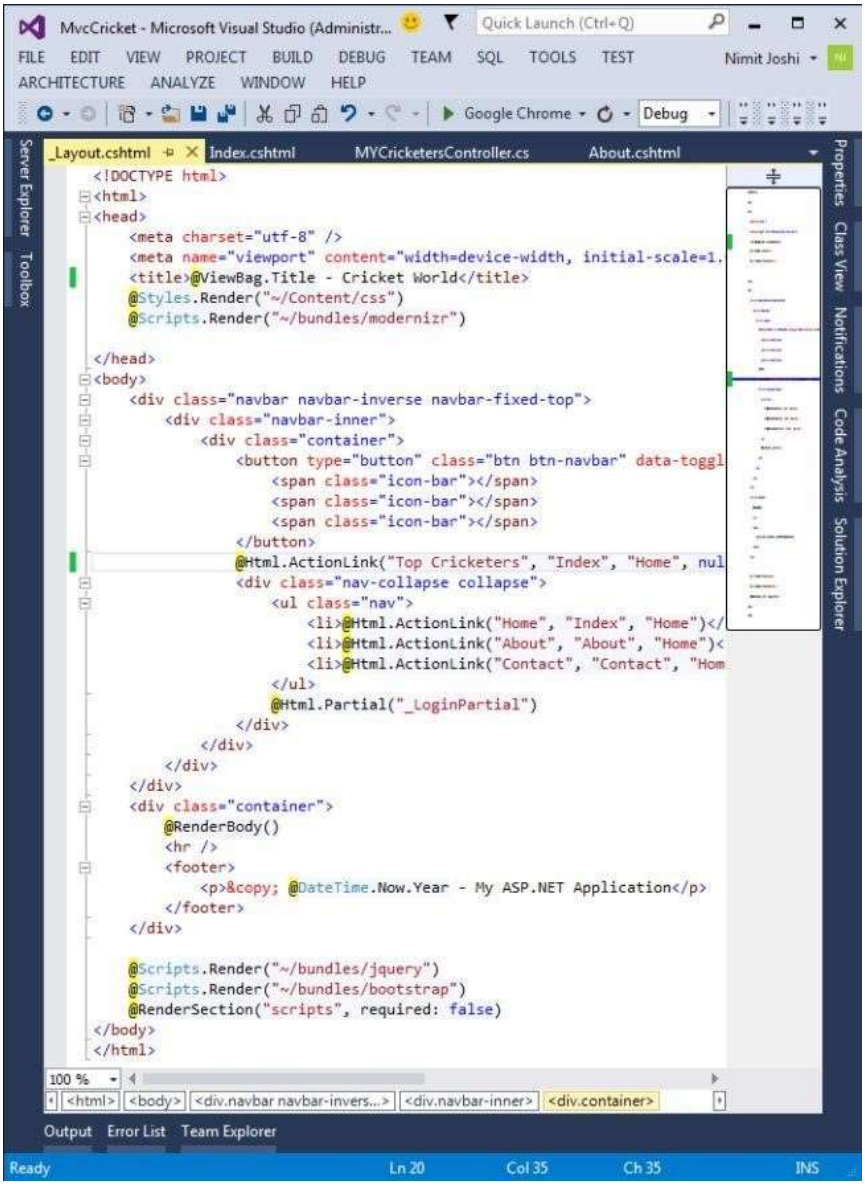
```
      routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

      routes.MapRoute(
          name: "Default",
          url: "{controller}/{action}/{id}",
          defaults: new { controller = "Home", action = "Index", id =
      UrlParameter.Optional }
);
```

Promena *pogleda* i Layout stranica

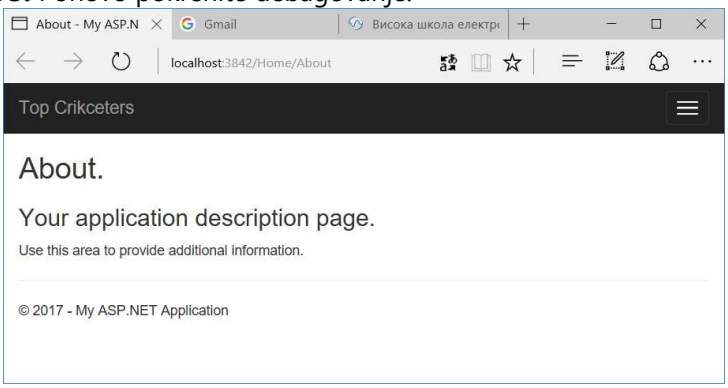
Korak 1: Označiti Layout stranicu u folderu **Shared**.



Korak 2: Na primer, ako želimo da promenimo ime aplikacije i naslov sa sopstvenim porebno je da promenimo ActionLink sa novim imenom u ovom fajlu.



Korak 3: Ponovo pokrenite debagovanje.

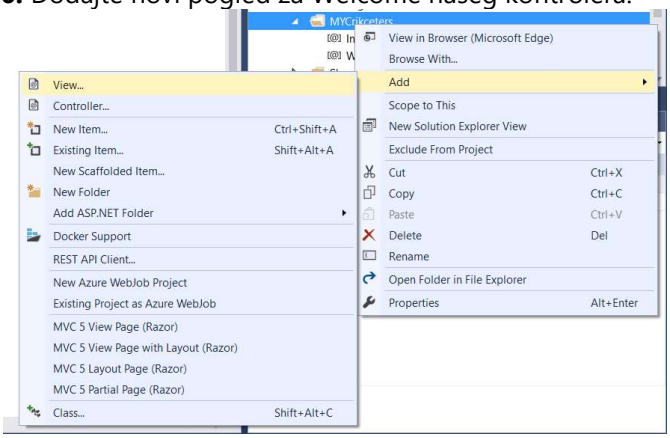


Korak 4: Zatim modifikujte kod kontrolera dodajući Welcome

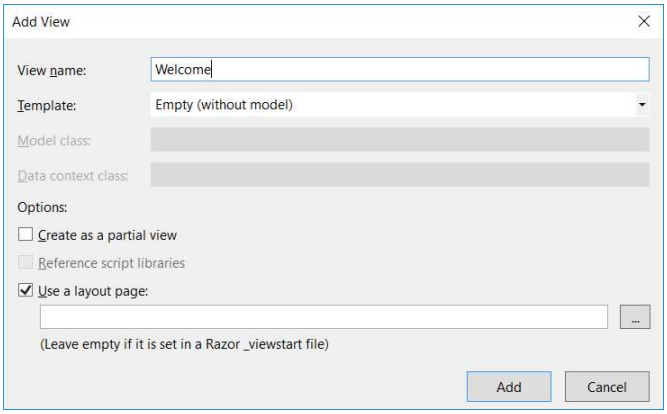
```
namespace prva_ASP_MVC_aplikacija_plus_Authent.Controllers
{
```

```
        public class MYCrikcetersController : Controller
        {
            public ActionResult Index()
            {
                return View();
            }

            public ActionResult Welcome()
            {
                ViewBag.Message = "Poruka iz Welcome";
                return View();
            }
        }
}
```

Korak 6: Dodajte novi pogled za Welcome našeg kontrolera.



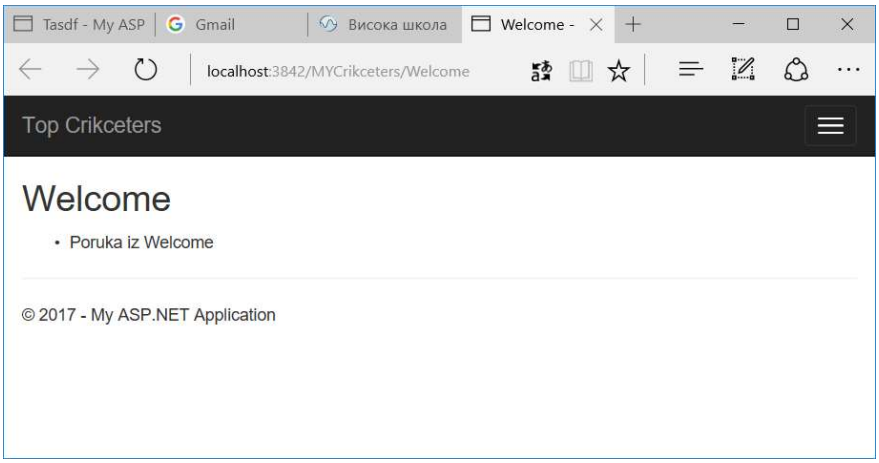
Korak 7: Dodajte ime pogleda.



Korak 8: Izmenite kod:

```
@{
    ViewBag.Title = "Welcome";
}
<h2>Welcome</h2>
<ul>
    <li>@ViewBag.Message</li>
</ul>
```

Korak 9: Pokrenite i pogledajte Welcome.



Pogled – engl. *View*

Za razliku od *file-based* web frejmvorka kao što su ASP.NET Web Forms ili PHP, pogledi nisu direktno dostupni. Nije moguće da ih direktno prikažemo u web čitaču već se njihov prikaz uvek renderuje od strane kontrolera. U nekim jednostavnijim slučajevima pogled zahteva malo ili nimalo informacija od kontrolera. Češće kontroler obezbeđuje informacije za pogled tako da prosleđuje objekte podataka koje nazivamo modelom. Pogled transformiše model u format spreman za prezentaciju korisniku.
Potrebno je da pogled ima odgovarajuće ime klase kontrolera.
Pogledajmo kreirani pogled za Home kontroler:

```
@{
    ViewBag.Title = "Home Page";
}

<div class="jumbotron">
    <h1>ASP.NET</h1>
    <p class="lead">ASP.NET is a free web framework for building great Web sites and Web applications
using HTML, CSS and JavaScript.</p>
    <p><a href="https://asp.net" class="btn btn-primary btn-lg">Learn more &raquo;</a></p>
</div>
<div class="row">
    <div class="col-md-4">
        <h2>Getting started</h2>
        <p>ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that
enables a clean separation of concerns and gives you full control over markup for enjoyable, agile
development.</p>
        <p><a class="btn btn-default" href="https://go.microsoft.com/fwlink/?LinkId=301865">Learn more
&raquo;</a></p>
    </div>
    <div class="col-md-4">
        <h2>Get more libraries</h2>
        <p>NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries
and tools in Visual Studio projects.</p>
        <p><a class="btn btn-default" href="https://go.microsoft.com/fwlink/?LinkId=301866">Learn more
&raquo;</a></p>
    </div>
    <div class="col-md-4">
        <h2>Web Hosting</h2>
        <p>You can easily find a web hosting company that offers the right mix of features and price for
your applications.</p>
        <p><a class="btn btn-default" href="https://go.microsoft.com/fwlink/?LinkId=301867">Learn more
&raquo;</a></p>
    </div>
</div>
```

A ovaj pogled se koristi u kontroleru:

```
public ActionResult Index()
{
    return View();
}
```

Osim metode Index pogledajte i ostale metode Home kontrolera. Videće na primer kod:

```
ViewBag.Message = "Your application description page.";
```

Očigledno se prosleđuje vrednost stringa svojstvu `Message` objekta `ViewBag` koji se zatim koristi u pogledima. ViewBag ima ograničenje i može biti korišćen za prenos male količine podataka.

Svaki folder kontrolera sadrži jedan view fajl za svaku akciju i fajl je imenovan na isti način kao i metod. Ovo obezbeđuje osnovu kako su pogledi pridruženi nekoj metodi akcije.
Naravno ovo je podrazumevana konvencija koja može biti promenjena. Na primer ako želimo da Index akcija renderuje drugi pogled jednostavno promeniti metodu Index i proslediti drugi pogled, na primer, umesto Index pogleda možemo proslediti pogled About:

```
public ActionResult Index()
{
    return View("About");
}
```

Može se pozvati podled iz potputno drugačijeg foldera. U ovom slučaju je potrebno navesti celu putanju

```
public ActionResult Index()
{
    return View("~/Views/Shared/Error.cshtml");
}
```

Zamislite zadatak da u pogledu prikažemo listu stavki koje ćemo generisati u kodu.
Kreirajmo najpre klasu Album za čuvaje podataka, koja je sastavljena od set i get metoda i sa samo dva polja. Album je klasa smeštena u folderu Model:

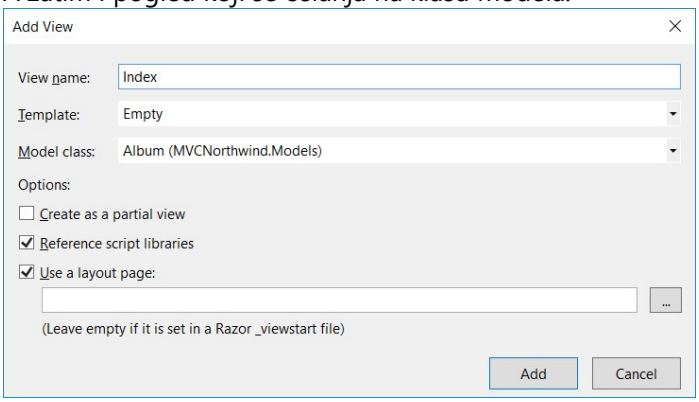
```
      public class Album
      {
          public int Id { get; set; }
          public string Name { get; set; }
}
```

Zatim dodajte AlbumListControler kao kontroler odnosno AlbumList pogled u folderu View I to na sledeći način, prvo kontroler:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace MVCNorthwind.Controllers
{
    public class AlbumListController : Controller
    {
        public ActionResult Index ()
        {
            Random rnd = new Random();
            var albums = new List<Models.Album>();
            for (int i = 0; i < 10; i++)
            {
             albums.Add(new Models.Album { Name = "Product " + rnd.Next(), Id = i + 1 });
            }
            ViewBag.Albums = albums;
```

```
            return View();
        }
    }
}
```

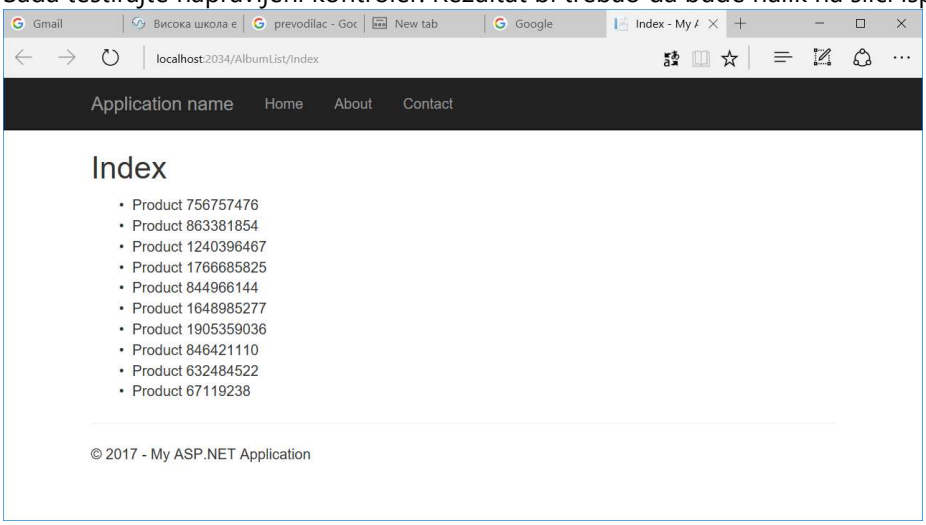
A zatim I pogled koji se oslanja na klasu modela.



Zatim izmenite pogled po uzoru na kod koji je ispod prikazan:

```
@using MVCNorthwind.Models;
@model MVCNorthwind.Models.Album

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>

<ul>
    @foreach (Album a in ViewBag.Albums)
    {
        <li>@a.Name</li>
    }
</ul>
```

Sada testirajte napravljeni kontroler. Rezultat bi trebao da bude nalik na slici ispod:



Inače, uključivanje imenskih prostora možete izvesti i iz fajla web.config (obratiti pažnju na prikaz IntellySense)

```
    <system.web.webPages.razor>
      <host factoryType="System.Web.Mvc.MvcWebRazorHostFactory, System.Web.Mvc,
    Version=5.2.3.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35" />
      <pages pageBaseType="System.Web.Mvc.WebViewPage">
        <namespaces>
          <add namespace="System.Web.Mvc" />
          <add namespace="System.Web.Mvc.Ajax" />
          <add namespace="System.Web.Mvc.Html" />
          <add namespace="System.Web.Optimization"/>
          <add namespace="System.Web.Routing" />

          <add namespace="AspNorthwind.Models" />

          <add namespace="AspNorthwind" />
        </namespaces>
      </pages>
</system.web.webPages.razor>
```

Upotreba **ViewBag** objekta je potpuno analogna upotrebi ViewDataDictiorary klase. To je specijalizovana klasa tipa rečnika namenjena skaldištenju podataka za prikaz. Zapravo ViewBag je omotač oko ove klase. Na primer:
ViewData["CurrentTime"] = DateTime.Now; I ovo je potpuno analogno sa
ViewBag.CurrentTime = DateTime.Now;
Kod je mnogo lakši za upotrebu i češće se koristi ako se koristi ViewBag objekat. Međutim postoji izuzetak kada to nije moguće. Na primer:
ViewData["Key With Spaces"]
Ova vrednost ne može da se adresira na isti način preko ViewBag objekta..

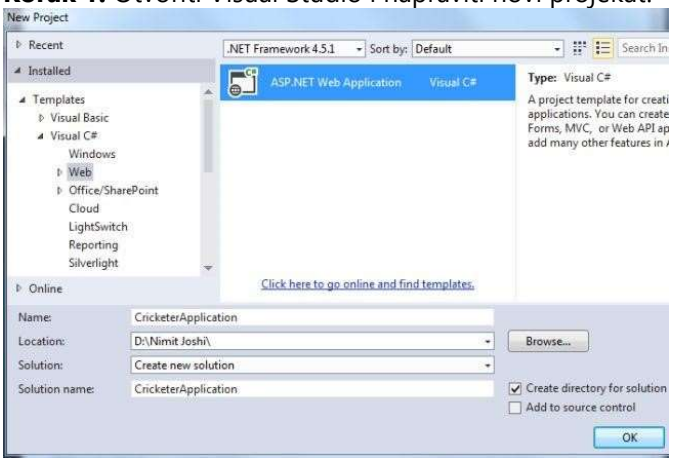
EF i razvoj tipa „Database First“

Ovde će biti objašnjeno povezivanje sa bazom koristeći EF Databse First pristup. Koristeći Scaffolding, lako ćemo generisati kod za Create, Read, Update, Delete (CRUD) operacije i to automatizovano.

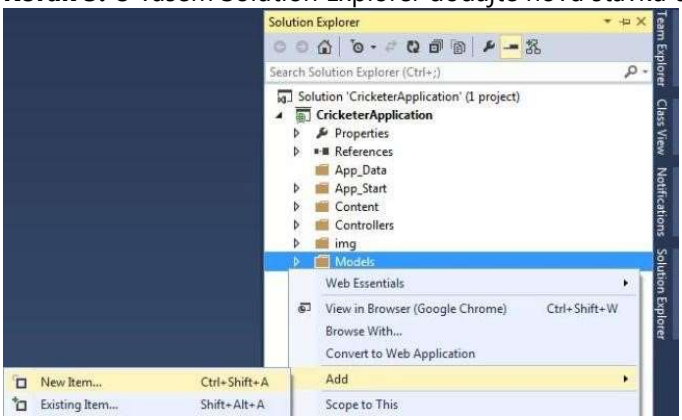
Pridruživanje baze aplikaciji

Web aplikacija u MVC

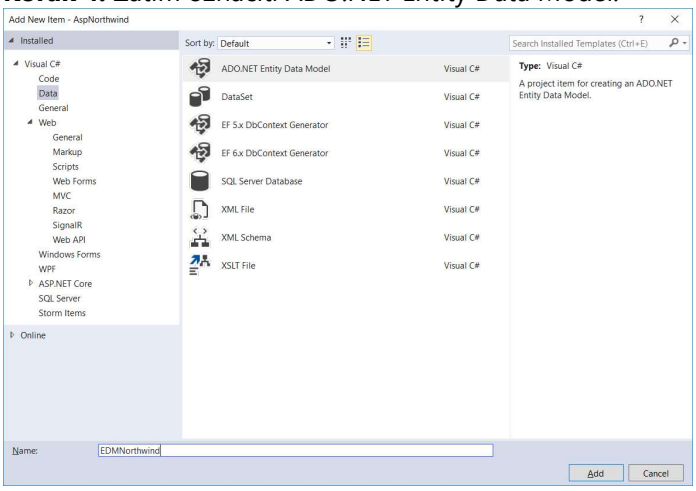
Korak 1: Otvoriti Visual Studio i napraviti novi projekat.

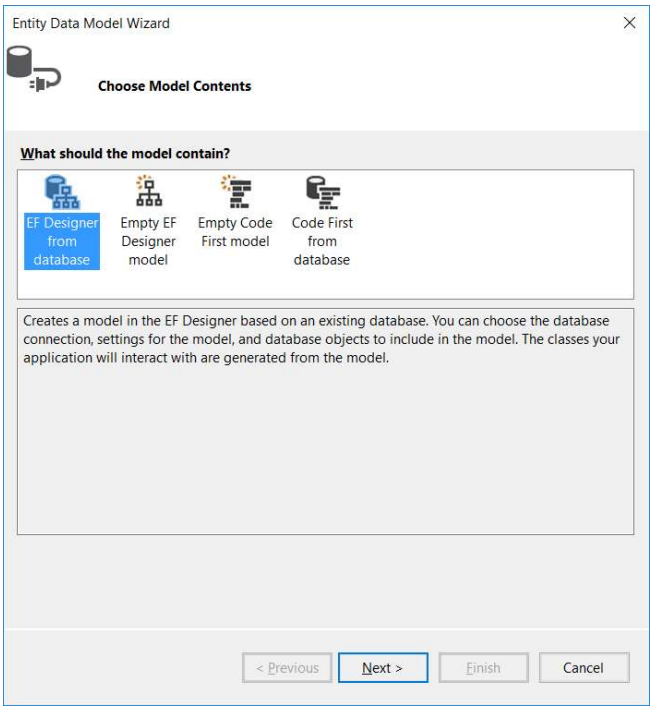


Korak 2: Označiti MVC project šablon.
Korak 3: U vašem Solution Explorer dodajte novu stavku desnim klikom na Models pa onda New Item.

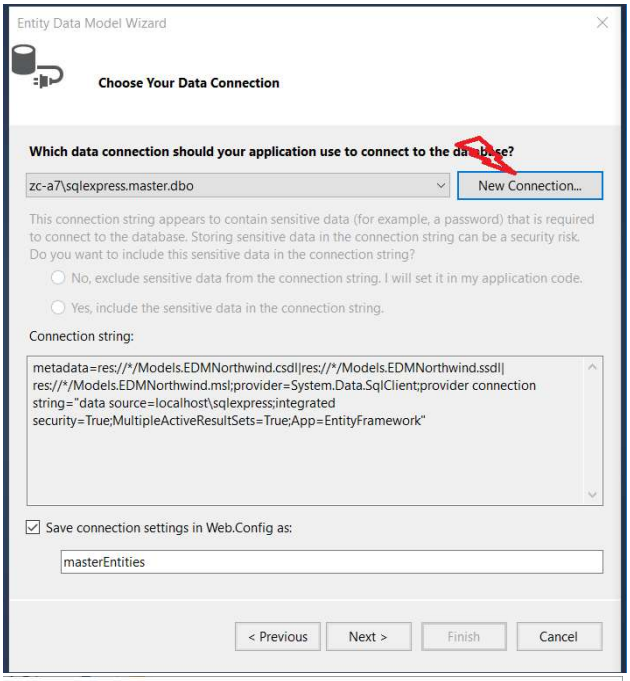


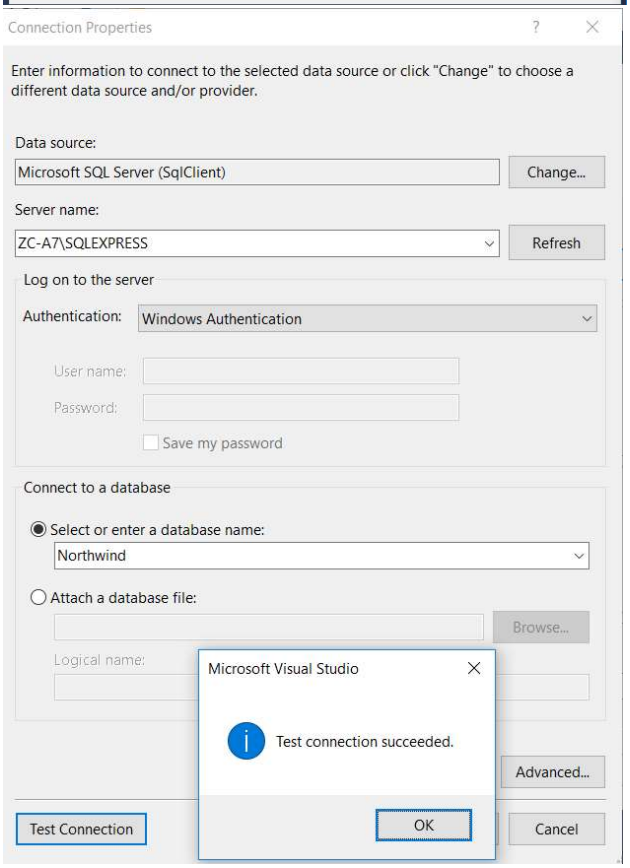
Korak 4: Zatim označiti ADO.NET Entity Data Model.





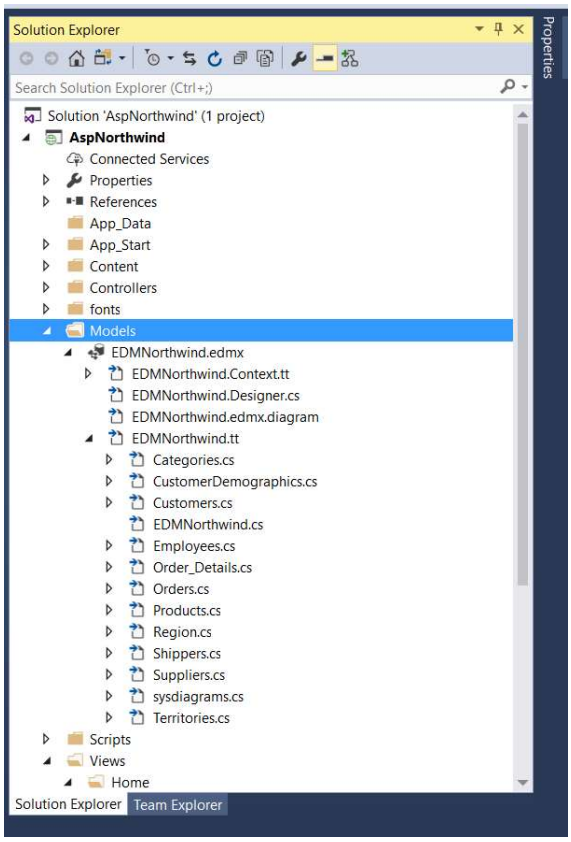
Korak 5: U narednom wizardu označiti postojeću konekciju ako postoji ili kreirati novu.





Korak 6: Ostatak postupka je poznat od ranije

Korak 7: Visual Studio automatski kreira Model.

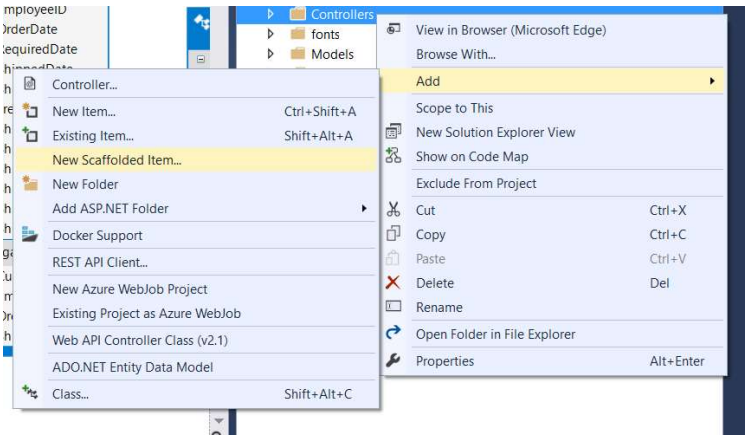


Nakon ovog postupka prelazimo da dodavanje kontrolera i pogleda i rad sa pogledima.

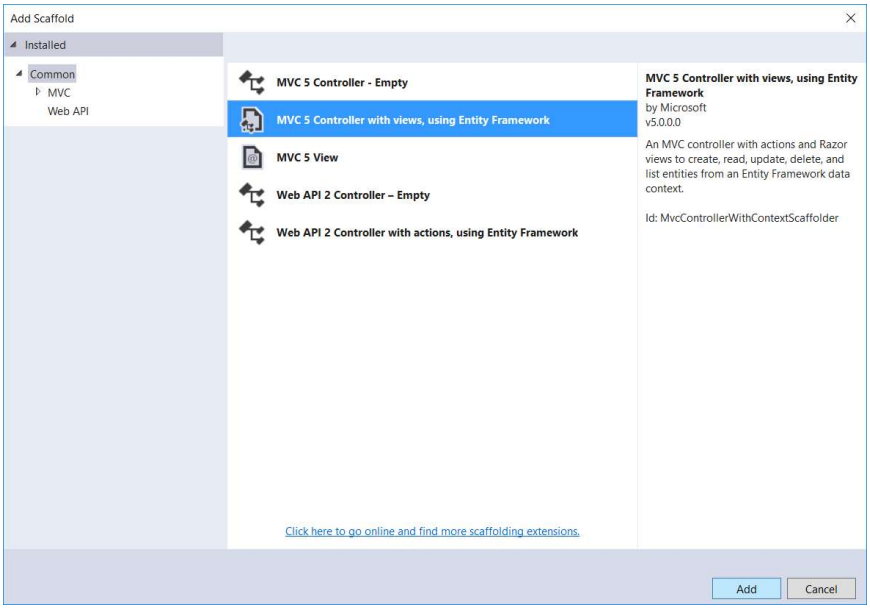
Dodavanje kontrolera i pogleda

MVC 5 može da kreira jedan kontroler koji odgovara pogledima koristeći Entity Framework 6. Potrebno je samo da koristimo Scaffolding. Pokrenimo sledeću proceduru.

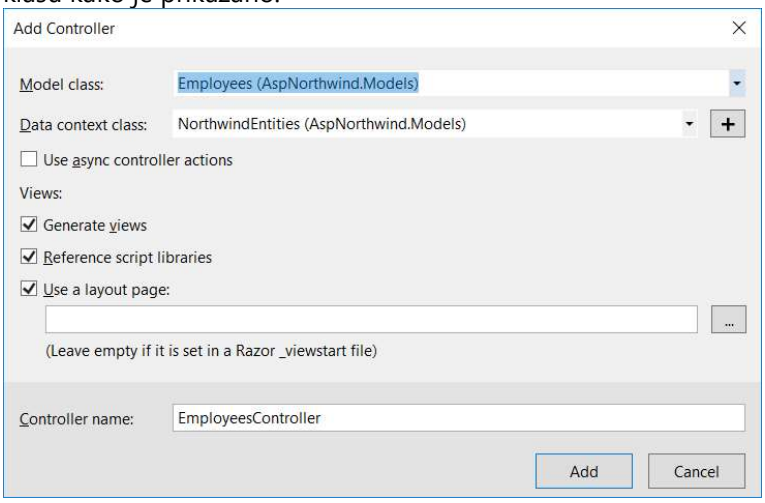
Korak 1: U Solution Exploreru desnim klik na folder Controller a zatim Add i birate opciju New Scaffolded Item



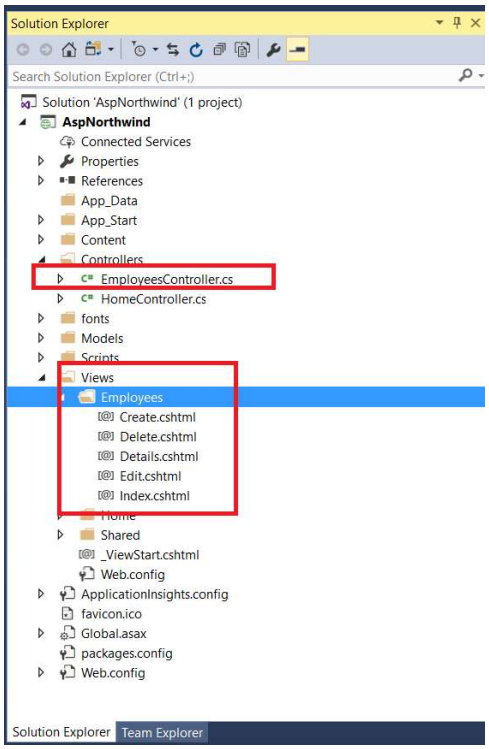
Korak 2: U sledećem čarobnjaku označiti kontroler sa opcijama pogleda kako je prikazano na slici:



Korak 3: U narednom čarobnjaku unesite ime kontrolera i odaberite odgovarajući model i DbContext klasu kako je prikazano:



Obratite pažnju da će biti prijavljena greška ako niste preveli vaš projekat.
Nakon dodavanja kontrolera VS automatski kreira EmployeesController i odgovarajući pogled:

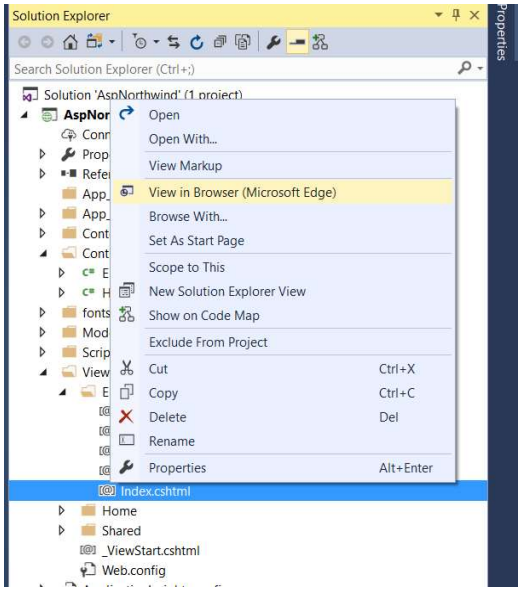


Rad sa pogledima

Sada su vaš controller i pogled spremni. Možete kreirati poglede za vaš web čitač. Za ovo pratite sledeću proceduru.

Create i Read operacije**: Employee**

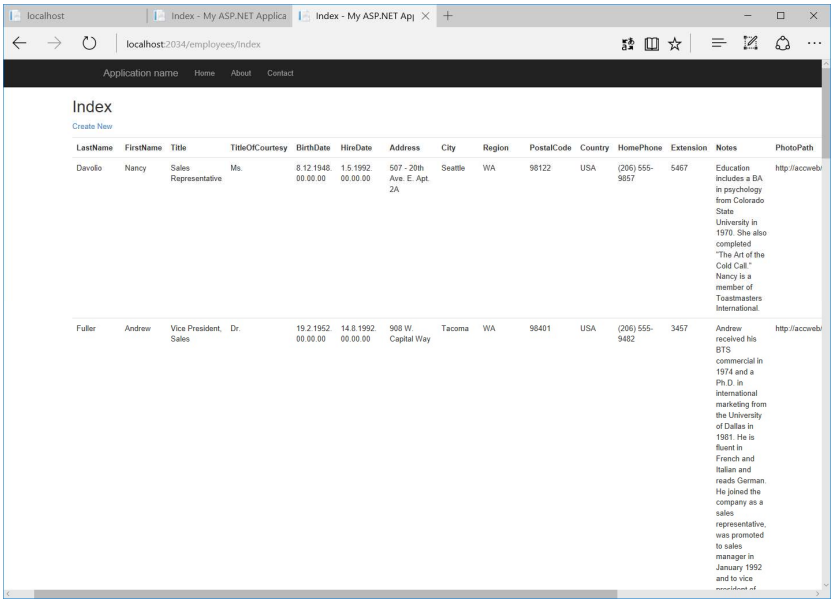
Korak 1: Označiti fajl Index.cshtml u pogledu Employees i otvriti ga u web čitaču preko konteksnog menija



Korak 2: Testirajte kreiranje novog radnika klikom na Create New. Unesite neke radnike. Izbacite kolonu Photo.

Employees Details: Edit & Delete operacije

Korak 1: Klik na Edit da bi pogledali informacije



Modifikacija aplikacije i validacija

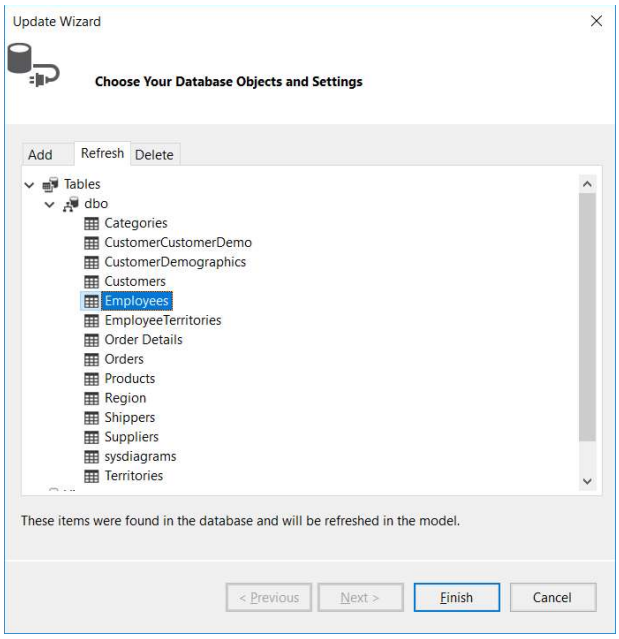
Obradićemo 3 celine:

- Modifikacija baze i aplikacije
- Data Annotation
- View doterivanje

Modifikacija baze i aplikacije

Korak 1: Promenite vašu bazu. Recimo da uvodimo novoPolje u tabeli Employees.

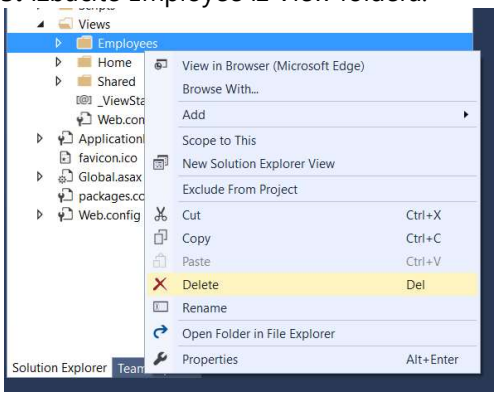
Korak 3: Promenu zapamtite i važno je da ne bude u koliziji sa već postojećim podacima.
Korak 4: Označiti vaš *Empolyees* Model preko fajla *EDMNorthwind.edmx* fajl. Desnim klikom na radnu površ i označite "Update Model from Database".



Korak 4: Sačuvaj fajl i prevedite aplikaciju.

Da bi se novo polje pojavilo u prikazu postoje dve opcije; koristeći Re-Scaffolding ili ručno odati svojstvo u pogledu. Uradićemo Re-Scaffolding.

Korak 5: Izbacite Employee iz View foldera.



Korak 6: Zatim dodajtne Scaffold ponovo iz foldera Controller *EmployeesController*. Klik na OK za zamenu postojećeg fajla.

Korak 7: Otvoriti *Index.cshtml* fajl u web čitaču i testirajte kreiranje novog reda.